Amendments to the Claims

1. (Currently Amended) A general computer network controller, coupled to ~~preferably operative in~~ a system area network, said controller comprising: ~~including~~

a data buffer configured to handle one or more ~~handling~~ payloads; ~~and~~

a fully associative context block configured to hold a plurality of last recently used contexts to provide a dynamic resource allocation scheme reflecting run time situations; and

a dedicated, programmable micro sequencer tightly coupled to said context block and configured to: ~~handling~~

control said context block; and

handle control flow and ~~being capable of running different~~ process multiple types of network packets and protocols; ~~, being~~

wherein said micro sequencer is packet format independent and network independent; and

~~, wherein said micro sequencer is tightly coupled to a fully associative context block for control thereof, said context block being operative to hold a number of last recently used contexts to provide a dynamic resource allocation scheme reflecting run time situations,~~

wherein ~~substantial parts of~~ said contexts are ~~being~~ updated by said micro sequencer, by an inbound scheduler and by a network protocol engine.

2. (Currently Amended) The computer network controller of claim 1, further comprising: ~~wherein said micro sequencer is operative to control~~

a scalable memory array configured ~~which can be used~~ as a table for Inbound address mapping of registered memory and access protection, and further configured as a means for keeping context information about all active channels.

3. (Currently Amended) The computer network controller of claim 1, wherein said fully associative context block couples ~~constitutes a connection between~~ said inbound scheduler and said network protocol engine, thereby facilitating an ability of ~~giving~~ said network controller ~~the ability~~ to pipeline tasks and execute in parallel.

4. (Currently Amended) The computer network controller of claim 3, wherein:

said context block is configured for dynamic allocation of ~~operative to have~~ contexts ~~dynamically allocated~~ between inbound remote direct memory access, inbound remote memory access and outbound remote memory access~~,~~;

~~wherein~~ two upper contexts are ~~nevertheless being~~ reserved for locally driven remote direct memory access,; and

said context block is configured to store ~~containing~~ information including one or more of the following events:

- expected sequence number of a ~~the~~ next packet for sequence checking,
- input gathering size in order to optimize use of an attached bus,
- packet type defined by the network for a specific virtual channel,
- accumulated message cyclic redundancy check for data integrity,
- source addresses,
- destination addresses,
- mapping for remote direct memory access operations,
- dedicated flags ~~like page crossing~~ to facilitate ~~do~~ new mapping,
- word count zero detection, and
- ~~as well as~~ protection tag check~~,~~; and

wherein said ~~all these information~~ events:

are received from said inbound scheduler, said micro sequencer and said network protocol engine;

are ~~to be~~ synchronized by said context block; and

are used by said micro sequencer to invoke, restart, switch or terminate a thread immediately.

5. (Currently Amended) The computer network controller of claim 1, wherein:

said micro sequencer is further configured ~~operative~~ to control said network protocol engine; ~~which in its turn is operative~~

said network protocol engine is configured to perform link injection control, based on feedback from a link layer and ~~as well as~~ intervention from an operative system,; and

said network protocol engine is further configured ~~being operative~~ to schedule packets to the network.

6. (Currently Amended) The computer network controller of claim 1, wherein said inbound scheduler is configured ~~operative~~ to decode, schedule and invoke running tasks or allocate new tasks, based on:

i) packets received from the network,

ii) memory mapped operations received from a bus attachment module,

iii) descriptors inserted in first-in, first-out work queues ~~fifos~~ by a user application, and

iv) tasks received from said context block.

7. (Currently Amended) In a system area network comprising a plurality of host channel adapters, a plurality of target channel adapters and a switching fabric, each ~~respective one of~~ said adapter comprising: ~~adapters being constituted by a computer network controller of the type defined in claim 1, together with a bus attachment module and a network link interface,~~

a data buffer configured to handle one or more payloads;

a fully associative context block configured to hold a plurality of last recently used contexts to provide a dynamic resource allocation scheme reflecting run time situations; and

a dedicated, programmable micro sequencer tightly coupled to said context block and configured to control said context block and handle control flow and process multiple types of network packets and protocols;

a bus attachment module; and

a network link interface;

wherein said micro sequencer is packet format independent and network independent, and wherein said contexts are updated by said micro sequencer, by an

inbound scheduler and by a network protocol engine,

a method for local and remote asynchronous completion control, the method comprising:

detecting a final packet of a message directed from a local node to a remote node, the final packet comprising:

an accumulated cyclic redundancy check covering the message; and

an address of a process completion queue on the remote node;

receiving the final packet at the remote node;

at the remote node:

performing an integrity check on the final packet;

signaling "receive complete" to the remote process completion queue; and

issuing a response to the final packet to the local node; and

at the local node, signaling "send complete" to a local process completion queue.

~~in which method as well accumulated message cyclic redundancy check as an address to a remote completion queue, e.g. at a target, are attached, by a said micro sequencer, to a last packet in a message to be sent from a sender, e.g. a host, to a receiver, e.g. a target, whereby, on reception of said packet at said receiver and checking for data integrity for the whole message by a target micro sequencer, "receive complete" is signaled directly from said target micro sequencer in the remote process completion queue, and simultaneously a response is made back to the sender, which will then signal "send complete" and status directly to a local process.~~

A2

8. (New) A protocol engine for a channel adapter configured to interface a system area network with a network node, the protocol engine comprising:

an inbound scheduler configured to schedule one or more of the following for each of a plurality of tasks: decoding, scheduling and invoking;

a multi-context micro sequencer configured to handle control flow for multiple communication channels between the network node and the system area network, wherein said multi-context micro sequencer is packet format independent and network independent;

a context block configured to store a set of least recently used contexts, wherein

each said context corresponds to one of the communication channels;

a data buffer configured to buffer payloads of packets for the multiple communication channels; and

a network protocol engine configured to schedule transmission of packets onto the system area network.

9. (New) The protocol engine of claim 8, wherein said multi-context micro sequencer is further configured to:

detect page boundary crossing and word count zero; and

perform an integrity check of a message, wherein the message comprises one or more packets.

10. (New) The protocol engine of claim 8, wherein said multi-context micro sequencer is further configured to perform integrated local and remote completion.

11. (New) The protocol engine of claim 8, wherein a subset of said contexts stored in said context block is reserved for outbound RDMA (Remote Direct Memory Access).

12. (New) The protocol engine of claim 11, wherein the remainder of said contexts in said set of contexts are dynamically allocated among inbound RDMA (Remote Direct Memory Access), inbound RMA (Remote Memory Access) and outbound RMA.

13. (New) The protocol engine of claim 8, wherein each said context stored in said context block comprises one or more of:

a source address;

a destination address;

RDMA operation mapping;

expected sequence number of a next packet;

an accumulated cyclic redundancy check; and

a set of dedicated flags for performing one or more of:

word count zero detection;

packet integrity checking;

sequence error checking;

protection tag checking; and

data buffer management.

14. (New) The protocol engine of claim 8, wherein said data buffer comprises a number of entries equivalent to the number of least recently used contexts stored in said context block.

15. (New) The protocol engine of claim 8, wherein said data buffer comprises:

multiple read ports; and

multiple write ports;

wherein said multiple read ports and multiple write ports facilitate processing of multiple tasks in parallel by the protocol engine.

16. (New) The protocol engine of claim 8, further comprising:

one or more work queues configured to store descriptors inserted by applications executing on the network node; and

an inbound scheduler configured to schedule processing of said descriptors.

17. (New) The protocol engine of claim 16, wherein said inbound scheduler is further configured to schedule:

receipt of a packet from the system area network;

a memory-mapped operation received from the network node; and

a task received from said context block.

18. (New) The protocol engine of claim 8, further comprising:

a first connection coupling the protocol engine to an internal bus of the network

node; and

a second connection coupling the protocol engine to the system area network.

19. (New) The protocol engine of claim 18, further comprising:

a third connection coupling the protocol engine to an address translation table;

wherein the address translation table is configured to:

maintain inbound address mapping; and

store context information not currently stored in said context block.

A2

20. (New) The protocol engine of claim 18, wherein the size of packets exchanged between the protocol engine and the network node differ from the size of packets exchanged between the protocol engine and the system area network.